Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 23.М04-мм

Разработка СПО прикладного уровня системы оплаты проезда российского производства на платных автодорогах

Пантелеймонов Андрей Радиевич

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
профессор кафедры системного программирования, д.ф.-м.н. А.Н. Терехов

Консультант:
Инженер-исследователь ООО «ЛИС», к.т.н. А.Г. Шадрин

Санкт-Петербург
2024

Оглавление

Введение

Первые платные дороги в России появились уже довольно давно - впервые на трассе М4 в 1998 году. С тех пор появились проекты, такие как ЗСД, которые требуют оплаты на всём участке пути. Постоянная оплата через оператора - человека сильно замедляет поток автомобилей и сокращает пропускную способность автодороги. Именно поэтому так важно наличие рабочей и безотказной системы оплаты проезда без участия человека.

По аналогии с банковскими картами, которые есть у любого человека, была придумана Система автоматического сбора пошлины (EFC, Electronic Fee Collenction), для работы который необходимо наличие транспондера или OBU (On-Board Unit, "устройства на борту") (аналога банковской карты) на лобовом стекле машины и RSU (Road-Side Unit, устройства на дороге), которое бы принимало и осуществляло транзакцию (аналог банкомата). Не так давно крупнейшая фирма, осуществлявшая полный цикл оплаты проезда такой системы Norbit ушла из России.

Таким образом появилась потребность в продукте, состоящем как из Программного обеспечения, так и аппаратного обеспечения, которое способно в полной объёме заменить разработки ушедшей норвежской фирмы.

Исходя из этого запроса, компании «Mobil-group» [4] и «ЛИС» [5] взялись за эту задачу, причём ответственными за ПО стала первая компания, а за создание АО — вторая.

Планируется, что продуктом будет физическое устройство с установленным на нём ПО, способным осуществить полный цикл обмена информацией и оплату с проезжающего мимо автомобиля с установленным на нём транспондером.

1 Постановка задачи

Цель работы — реализовать прикладной уровень выделенной радиосвязи ближнего действия

Задачи на осенний семестр:

- Сделать обзор имеющейся документации предыдущих производителей на российском рынке
- Проанализоровать соответствующие стандарты необходимые для разработки системы
- Реализовать выдачу информации в режиме реального времени для информирования оператора о текущем статусе в терминах описанных в документах протоколов
- Реализовать логирование работы системы для анализа ошибок и сбора статистики
- Приступить к реализации обмена информации между канальным и прикладным уровнями

Задачи на весенний семестр:

- Закончить реализацию обмена информации между канальным и прикладным уровнями
- Реализовать обмен информацией с биллинговой системой
- Провести апробацию продукта

2 Обзор

2.1 Обзор предметной области

2.1.1 Архитектура DSRC

Архитектура Выделенной Радиосвязи Ближнего действия (DSRC — Dedicated Short-Range Communication) состоит из нескольких компонентов:

- OBU (On-Board Unit) — транспондер или устройство, которое находится на транспортном средстве, служащее картой оплаты
- RSU (Road-Side Unit) — устройство, которое расположено на пути взимания оплаты проезда и осуществляющее обмен данными не только с транспондером, но и с внутренними сервисами, а также системой биллинга
- Application Layer Core — ядро прикладного уровня, которое отвечает за обмен информацией с канальным уровнем и приложением, а также ядром другого устройства (OBU или RSU)
- Data Link Layer (L2) — канальный уровень
- Physical Layer (L1) — физический уровень

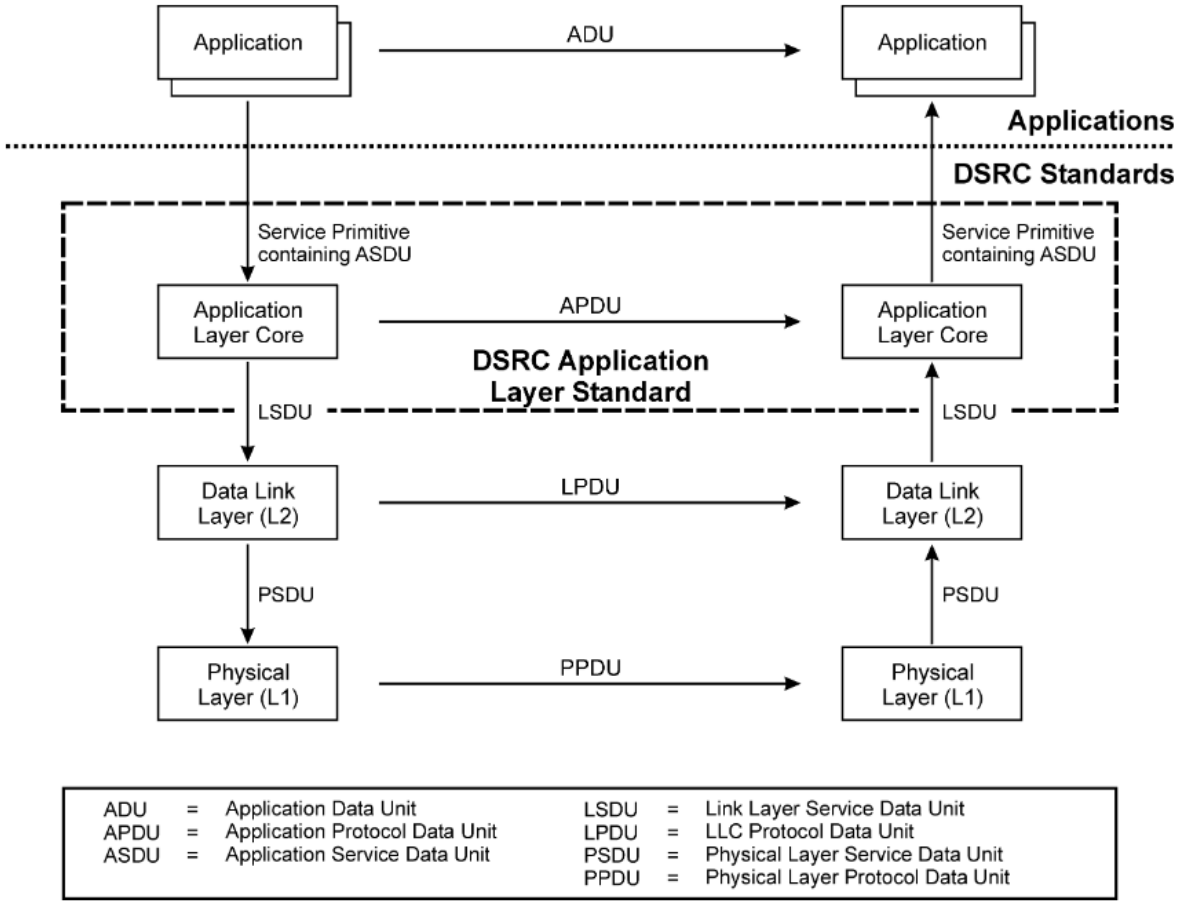


Рис. 1: Архитектура стека DSRC, скриншот взят из [1]

2.1.2 Архитектура прикладного уровня DSRC

Архитектура прикладного уровня DSRC состоит из нескольких компонентов:

- I-Kernel — ядро инициализации, которое отвечает за инициализацию обмена информацией между RSU и OBU.
- T-Kernel — трансферное ядро, задача которого — обмен данными между канальным и прикладным уровнем и прикладным уровнем и приложением, а также с прикладным уровнем другой сущности (OBU или RSU)
- B-Kernel — ядро широколосной передачи, которое должно реализовывать сбор, широкую передачу и распространение информа-

ции для различных применений путём обмена через широковещательный пул

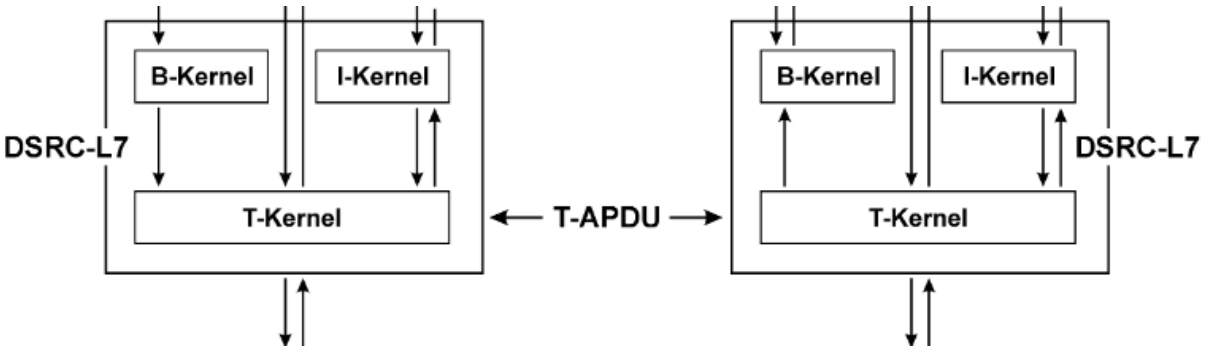


Рис. 2: Архитектура прикладного уровня DSRC

2.1.3 Обзор этапов обмена данными в ходе транзакции

2.1.3.1 BST

Первым этапом является BST - Beacon Service Table - данные, которые непрерывно посылает RSU по направлению к дороге, ожидая получения ответа от какого-либо OBU. На изображении ниже приведена его структура

Table B.11 — Initialisation request (BST) frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 1110	Start Flag
2	Broadcast LID	1111 1111	Link address for broadcast
3	MAC control field	1010 0000	The frame contains a command LPDU
4	LLC control field	0000 0011	UI command
5	Fragmentation header	1xxx x001	No fragmentation. PDU no shall never be set to 0000_2 or 0001_2.
6	BST SEQUENCE {	1000	INITIALISATION.request
	OPTION indicator	0	NonmandApplications not present.
	BeaconId SEQUENCE { ManufacturerId INTEGER (0..65535)	000	Manufacturer identifier- Example 1 (=Kapsch). See ISO 14816.
7		0000 0000	Register at www.tc278.eu/index.php/14816-registers for value assignment.
8		0000 1	
	IndividualId INTEGER (0..134217727)	000	27 bit ID available for manufacturer. Example: Id=1052_{10}
9		0000 0000	
10		0000 0100	
11	}	0001 1100	
12	Time INTEGER(0..4294967295)	0100 0001	32 bit UNIX real time. Example: 1103790512_{10}
13		1100 1010	
14		1000 0001	
15		1011 0000	
16	Profile INTEGER (0..127,...)	0000 0000	No extension, Profile. Example : Profile = 0

Рис. 3: Битовое представление BST

2.1.3.2 VST

Ответом на BST является VST (Vehicle Service Table) - данные, которые ”зашиты” в транспондер. Например, страна регистрации транспондера, ключи для алгоритма 3DES для дальнейших этапов транзакции.

B.4.2.4 Initialisation response (VST)

Table B.14 — Initialisation response (VST) frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 1110	Start Flag
2	Private LID	xxxx xxx0	Link address of a specific OBE
3		xxxx xxx0	
4		xxxx xxx0	
5		xxxx xxx1	
6	MAC control field	1100 0000	The frame contains a command LPDU
7	LLC control field	0000 0011	UI command
8	Fragmentation header	1xxx x001	No fragmentation. PDU no shall never be set to 0000_2 or 0001_2.
9	VST SEQUENCE {	1001	INITIALISATION.response
	Fill BIT STRING (SIZE(4))	0000	Set to 0
10	Profile INTEGER (0..127,...)	0000 0000	No extension, Profile. Example : 0_{10}
11	Applications SEQUENCE (SIZE((0..127,...)) OF {	0000 0010	No extension, 2 applications
12	SEQUENCE { OPTION indicator	1	EID present
	OPTION indicator	1	Parameter present
	AID DSRCApplicationEntityID	00 0001	No extension, AID = 1, EFC
13	EID	0000 0010	Associated with a context mark. Example : 2_{10}
14	Parameter CONTAINER {	0000 0010	No extension, Container Choice = 2_{10}, Octet string
15		0000 0110	No extension, octet string length = 6_{10}
16	EFC-ContextMark SEQUENCE {		

Table B.14 *(continued)*

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
	ContractProvider SEQUENCE {		
	CountryCode BIT STRING (SIZE(10))	CC11 CCOO	10 bit country code according to ISO 3166 with ITA2
17		11	Binary encoding based on ISO 14816. Example : NO
	IssuerIdentifier INTEGER (0..16383) }	OO CCCO	14 bits issuer identifier. Example : 2_{10}
18		CCOO CC1O	
19	TypeOfContract OCTET STRING (SIZE(2))	CCOO CCOO	Type of contract. Example : 1_{10}
20		CCOO CCO1	
21	ContextVersion INTEGER (0..127..) } } }	CCOO CC1O	No extension, context version. Example : 2_{10}
22	SEQUENCE { OPTION indicator	1	EID present
	OPTION indicator	1	Parameter present
	AID DSRCApplicationEntityID	OC CO01	No extension, AID = 1, EFC
23	EID	COOC C101	Associated with a context mark. Example : 5_{10}
24	Parameter CONTAINER {	COOC C01C	No extension, Container Choice = 2_{10}, Octet string
25		0001 OOOC	No extension, octet string length = 16_{10}
26	EFC-ContextMark SEQUENCE {		
	ContractProvider SEQUENCE {		
	CountryCode BIT STRING (SIZE(10))	101C C10C	10 bit country code according to ISO 3166 with ITA2 binary
27		OC	Encoding based on ISO 14816. Example : SE
	IssuerIdentifier INTEGER (0..16383) }	CO OOCO	14 bits issuer identifier. Example : 1_{10} (Öresundskonsortiet)
28		CCOO CCO1	
29	TypeOfContract OCTET STRING (SIZE(2))	CCOO CCOO	Type of contract. Example : 2_{10}
30		CCOO CC1O	
31	ContextVersion INTEGER (0..127..) }	CCOO CCO1	No extension, context version. Example : 1_{10}
32	CONTAINER {	CCOO CC1O	No extension, Container Choice = 2_{10}, Octet string
33		CCOO CC1O	No extension, octet string length = 2_{10}
34	AC_CR-Reference SEQUENCE { AC-MasterKeyRef Int1,	CCOO CCO1	AC_CR-Reference to, consisting of AC_CR-MasterKeyRef and AC_CR-Diversifier, used for the computation of AC_CRKey and
35	AC_CR-Diversifier Int1 } }	CCOO CCO1	AC_CR.
36	CONTAINER {	CCOO CC1O	No extension, Container Choice = 2_{10}, Octet string
37		CCOO C10O	No extension, octet string length = 4_{10}
38	RndOBE Int4	CCOO CCOO	Random Number (nonce) used together with AC_CRKey to calculate AC_CR. Example : 640_{10}
39		CCOO CCOO	

Table B.14 *(continued)*

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
40		COOC C01C	
41	} } } }	1OOC OOOC	
42	ObeConfiguration SEQUENCE {		
	OPTION indicator	1	ObeStatus present
	EquipmentClass INTEGER (0..32767)	CCC OOCC	Example : 3_{10}
43		COOC C011	
44	ManufacturerId INTEGER (0..65535)	COOO COOO	Manufacturer identifier. See ISO 14816 Register at
45		COOO CO1O	www.tc278.eu/index.php/14816-registers for value assignment. Example : 2_{10}.
46	ObeStatus INTEGER(0..65535)	OOOO OO11	Example : 768_{10}
47	} }	COOO COOO	
48	FCS	xxxx xxxx	Frame check sequence
49		xxxx xxxx	
50	FLAG	C111 111C	End Flag

Рис. 4: Битовое представление VST

2.1.3.3 Presentation Request

После получения VST RSE в том числе получает ключ для вычисления

значения используя алгоритм шифрования, после чего запрашивает уже более детальную информацию с использованием своего результата дешифровки данных, полученных от транспондера. К этой информации, например, относится класс автомобиля, максимально возможная масса и т.д.

B.4.3.1 Presentation request

Table B.15 — Presentation request frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 111C	Start Flag
2	Private LID	xxxx xxx0	Link address of a specific OBE
3		xxxx xxx0	
4		xxxx xxx0	
5		xxxx xxx1	
6	MAC control field	1C10 sCC0	The frame contains a command LPDU
7	LLC control field	n111 C111	Polled ACn command, n bit
8	Fragmentation header	1xxx xCC1	No fragmentation. First service of chain.
9	ACTION.request SEQUENCE {	0000	ACTION.request (GET_STAMPED.request)
	OPTION indicator	1	AccessCredentials present
	OPTION indicator	1	ActionParameter present
	OPTION indicator	0	IID not present
	Mode BOOLEAN	1	Mode = TRUE, Response expected
10	EID INTEGER(0..127,...)	0000 0101	No extension, Element EID, uniquely related to a Context mark within the OBE. Example : 5_{10}
NOTE VehicleSpecificCharacteristics, VehicleDimensions and Vehicle Axles are not included in the examples in this table.			

Table B.15 *(continued)*

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
11	ActionType INTEGER(0..127,...)	0000 0000	No extension, Action type = 0, GET_STAMPED. request
12	AccessCredentials OCTET STRING {	0000 0100	No extension, octet string length = 4_{10}
13	AC_CR	aaaa aaaa	Access credentials calculated by RSE using RndOBE and the
14		aaaa aaaa	Access Credentials Key AC_CRKey.
15		aaaa aaaa	
16	}	aaaa aaaa	
17	ActionParameter CONTAINER {	0001 0001	No extension, Container Choice = 17_{10}, GetStampedRq
18	AttributeIdList SEQUENCE (SIZE(0..127,...)) OF { INTEGER (0..127,...) AttributeId	0000 0001	No extension, number of attribute IDs = 1
19	PaymentMeans }	001C 0000	No extension, AttributeId = 32_{10}, PaymentMeans
20	Nonce OCTET STRING {	0000 0100	No extension, octet string length = 4_{10}
21	RndRSE	rrrr rrrr	Random number from RSE, containing SessionTime, needed to calculate OperatorAuthenticator
22		rrrr rrrr	
23		rrrr rrrr	
24	}	rrrr rrrr	
25	KeyRef_Op(h) } }	xxxx xxxx	h = Reference to AuKey_Op used for the computation of Operator Authenticator.
26	Fragmentation header	1xxx xC01	No fragmentation. Same PDU no as before (concatenation).
27	GET.request SEQUENCE {	C110	GET.request
	OPTION indicator	1	AccessCredentials present
	OPTION indicator	C	IID not present
	OPTION indicator	1	AttributeIdList present
	Fill BIT STRING(SIZE(1))	C	Set to 0
28	EID INTEGER(0..127,...)	0000 0101	No extension, EID, Example : 5_{10}
29	AccessCredentials OCTET STRING {	0000 0100	No extension, octet string length = 4_{10}
30	AC_CR	aaaa aaaa	Access credentials calculated by RSE using RndOBE and the
31		aaaa aaaa	
32		aaaa aaaa	Access Credentials Key AC_CRKey.
33	}	aaaa aaaa	
34	AttributeIdList SEQUENCE (SIZE(0..127,...)) OF { INTEGER (0..127,...) AttributeId	0000 0110	No extension, number of attribute IDs = 6_{10}
35	VehicleLicencePlateNumber	0001 0000	No extension, AttributeId = 16_{10}, VehicleLicencePlateNr
36	VehicleClass	0001 0001	No extension, AttributeId = 17_{10}, VehicleClass
37	VehicleWeightLimits	0001 0100	No extension, AttributeId = 20_{10}, VehicleWeightLimits
NOTE VehicleSpecificCharacteristics, VehicleDimensions and Vehicle Axles are not included in the examples in this table.			

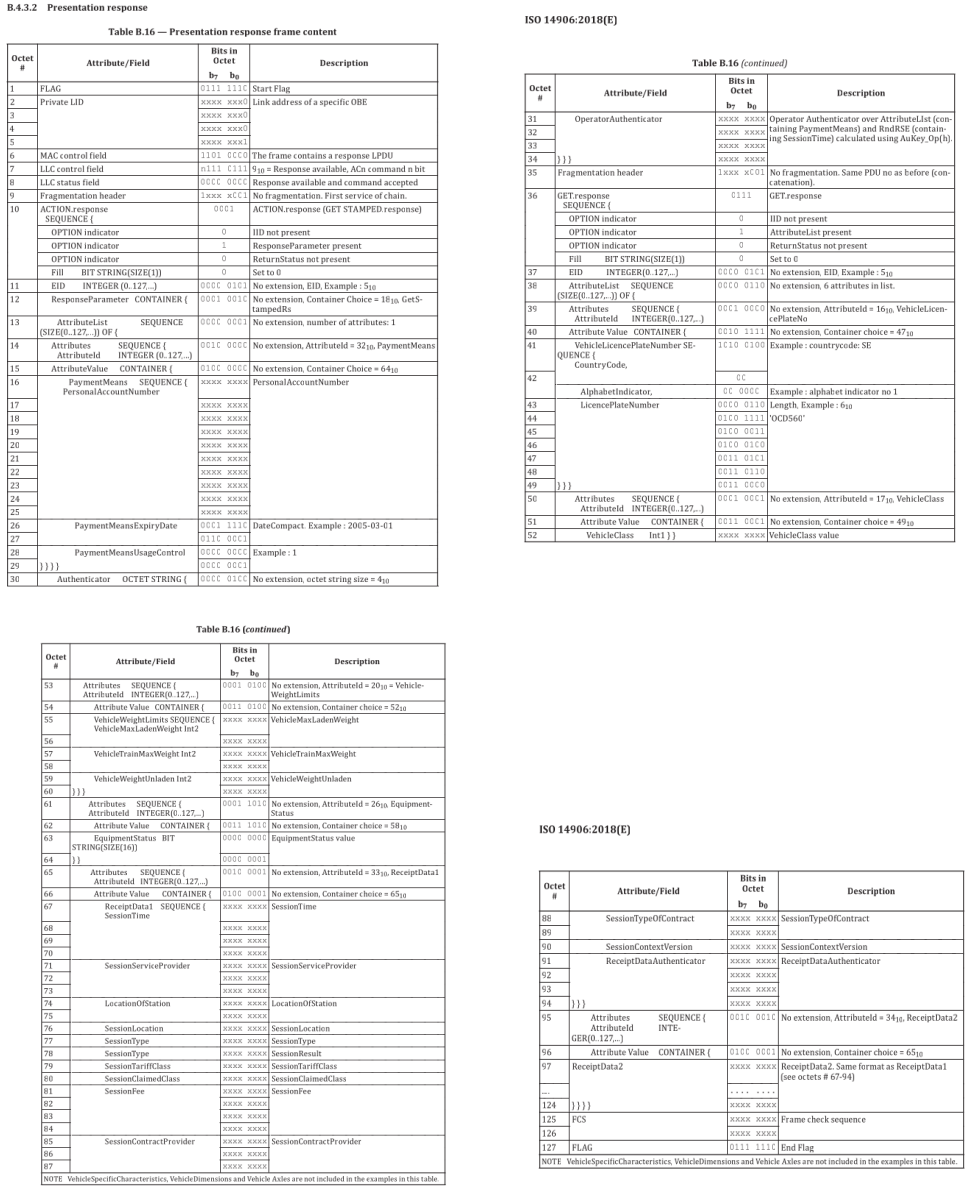
Table B.15 *(continued)*

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
38	EquipmentStatus	0001 101C	No extension, AttributeId = 26_{10}, EquipmentStatus
39	ReceiptData1	001C 0001	No extension, AttributeId = 33_{10}, ReceiptData1
40	ReceiptData2 } }	001C 001C	No extension, AttributeId = 34_{10}, ReceiptData2
41	FCS	xxxx xxxx	Frame check sequence
42		xxxx xxxx	
43	FLAG	0111 111C	End Flag
NOTE VehicleSpecificCharacteristics, VehicleDimensions and Vehicle Axles are not included in the examples in this table.			

Рис. 5: Структура полей Presentation Request

2.1.3.4 Presentation Response

Если вычисленные значения на стороне RSE совпали со значениями на OBU, то транспондер отпарвляет ответ с запрашиваемыми данными:



B.4.3.2 Presentation response

Table B.16 — Presentation response frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 111C	Start Flag
2	Private LID	xxxx xxx0	Link address of a specific OBE
3		xxxx xxx0	
4		xxxx xxx0	
5		xxxx xxx1	
6	MAC control field	1101 0CC0	The frame contains a response LPDU
7	LLC control field	n111 C111	9_{10} = Response available, ACn command n bit
8	LLC status field	0CCC 0CCC	Response available and command accepted
9	Fragmentation header	1xxx xCC1	No fragmentation. First service of chain.
10	ACTION.response SEQUENCE {	0001	ACTION.response (GET STAMPED response)
	OPTION indicator	0	IID not present
	OPTION indicator	1	ResponseParameter present
	OPTION indicator	0	ReturnStatus not present
	Fill BIT STRING(SIZE(1))	0	Set to 0
11	EID INTEGER (0..127,...)	0CCC 01C1	No extension, EID, Example : 5_{10}
12	ResponseParameter CONTAINER {	0001 001C	No extension, Container Choice = 18_{10}, GetStampedRs
13	AttributeList SEQUENCE {SIZE(0..127,...)} OF {	0CCC 0CC1	No extension, number of attributes: 1
14	Attributes SEQUENCE { AttributeId INTEGER (0..127,...)	001C 0CCC	No extension, AttributeId = 32_{10}, PaymentMeans
15	AttributeValue CONTAINER {	010C 0CCC	No extension, Container Choice = 64_{10}
16	PaymentMeans SEQUENCE { PersonalAccountNumber	xxxx xxxx	PersonalAccountNumber
17		xxxx xxxx	
18		xxxx xxxx	
19		xxxx xxxx	
20		xxxx xxxx	
21		xxxx xxxx	
22		xxxx xxxx	
23		xxxx xxxx	
24		xxxx xxxx	
25		xxxx xxxx	
26	PaymentMeansExpiryDate	0CC1 111C	DateCompact. Example : 2005-03-01
27		011C 0CC1	
28	PaymentMeansUsageControl	0CCC 0CCC	Example : 1
29	} } } }	0CCC 0CC1	
30	Authenticator OCTET STRING {	0CCC 01CC	No extension, octet string size = 4_{10}

Table B.16 (*continued*)

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
53	Attributes SEQUENCE { AttributeId INTEGER(0..127,...)	0001 010C	No extension, AttributeId = 20_{10} = VehicleWeightLimits
54	Attribute Value CONTAINER {	0011 010C	No extension, Container choice = 52_{10}
55	VehicleWeightLimits SEQUENCE { VehicleMaxLadenWeight Int2	xxxx xxxx	VehicleMaxLadenWeight
56		xxxx xxxx	
57	VehicleTrainMaxWeight Int2	xxxx xxxx	VehicleTrainMaxWeight
58		xxxx xxxx	
59	VehicleWeightUnladen Int2	xxxx xxxx	VehicleWeightUnladen
60	} } }	xxxx xxxx	
61	Attributes SEQUENCE { AttributeId INTEGER(0..127,...)	0001 101C	No extension, AttributeId = 26_{10}, EquipmentStatus
62	Attribute Value CONTAINER {	0011 101C	No extension, Container choice = 58_{10}
63	EquipmentStatus BIT STRING{SIZE(16)}	000C 000C	EquipmentStatus value
64	} }	000C 0001	
65	Attributes SEQUENCE { AttributeId INTEGER(0..127,...)	001C 0001	No extension, AttributeId = 33_{10}, ReceiptData1
66	Attribute Value CONTAINER {	010C 0001	No extension, Container choice = 65_{10}
67	ReceiptData1 SEQUENCE { SessionTime	xxxx xxxx	SessionTime
68		xxxx xxxx	
69		xxxx xxxx	
70		xxxx xxxx	
71	SessionServiceProvider	xxxx xxxx	SessionServiceProvider
72		xxxx xxxx	
73		xxxx xxxx	
74	LocationOfStation	xxxx xxxx	LocationOfStation
75		xxxx xxxx	
76	SessionLocation	xxxx xxxx	SessionLocation
77	SessionType	xxxx xxxx	SessionType
78	SessionType	xxxx xxxx	SessionResult
79	SessionTariffClass	xxxx xxxx	SessionTariffClass
80	SessionClaimedClass	xxxx xxxx	SessionClaimedClass
81	SessionFee	xxxx xxxx	SessionFee
82		xxxx xxxx	
83		xxxx xxxx	
84		xxxx xxxx	
85	SessionContractProvider	xxxx xxxx	SessionContractProvider
86		xxxx xxxx	
87		xxxx xxxx	

NOTE VehicleSpecificCharacteristics, VehicleDimensions and Vehicle Axles are not included in the examples in this table.

ISO 14906:2018(E)

Table B.16 *(continued)*

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
31	OperatorAuthenticator	xxxx xxxx	Operator Authenticator over AttributeLIst (containing PaymentMeans) and RndRSE (containing SessionTime) calculated using AuKey_Op(h).
32		xxxx xxxx	
33		xxxx xxxx	
34	} } }	xxxx xxxx	
35	Fragmentation header	1xxx xCC1	No fragmentation. Same PDU no as before (concatenation).
36	GET.response SEQUENCE {	0111	GET.response
	OPTION indicator	0	IID not present
	OPTION indicator	1	AttributeList present
	OPTION indicator	0	ReturnStatus not present
	Fill BIT STRING(SIZE(1))	0	Set to 0
37	EID INTEGER(0..127,...)	0CC0 01C1	No extension, EID, Example : 5_{10}
38	AttributeList SEQUENCE {SIZE(0..127,...)} OF {	0CC0 0110	No extension, 6 attributes in list.
39	Attributes SEQUENCE { AttributeId INTEGER(0..127,...)	00C1 0CC0	No extension, AttributeId = 16_{10}, VehicleLicencePlateNo
40	Attribute Value CONTAINER {	0C10 1111	No extension, Container choice = 47_{10}
41	VehicleLicencePlateNumber SEQUENCE { CountryCode,	1C10 C100	Example : countrycode: SE
42		0C	
	AlphabetIndicator,	0C 00CC	Example : alphabet indicator no 1
43	LicencePlateNumber	00C0 0110	Length, Example : 6_{10}
44		01C0 1111	'OCD560'
45		01C0 0C11	
46		01C0 01C0	
47		0C11 01C1	
48		0C11 C110	
49	} } }	0C11 0CC0	
50	Attributes SEQUENCE { AttributeId INTEGER(0..127,...)	00C1 00C1	No extension, AttributeId = 17_{10}, VehicleClass
51	Attribute Value CONTAINER {	0C11 00C1	No extension, Container choice = 49_{10}
52	VehicleClass Int1 } }	xxxx xxxx	VehicleClass value

ISO 14906:2018(E)

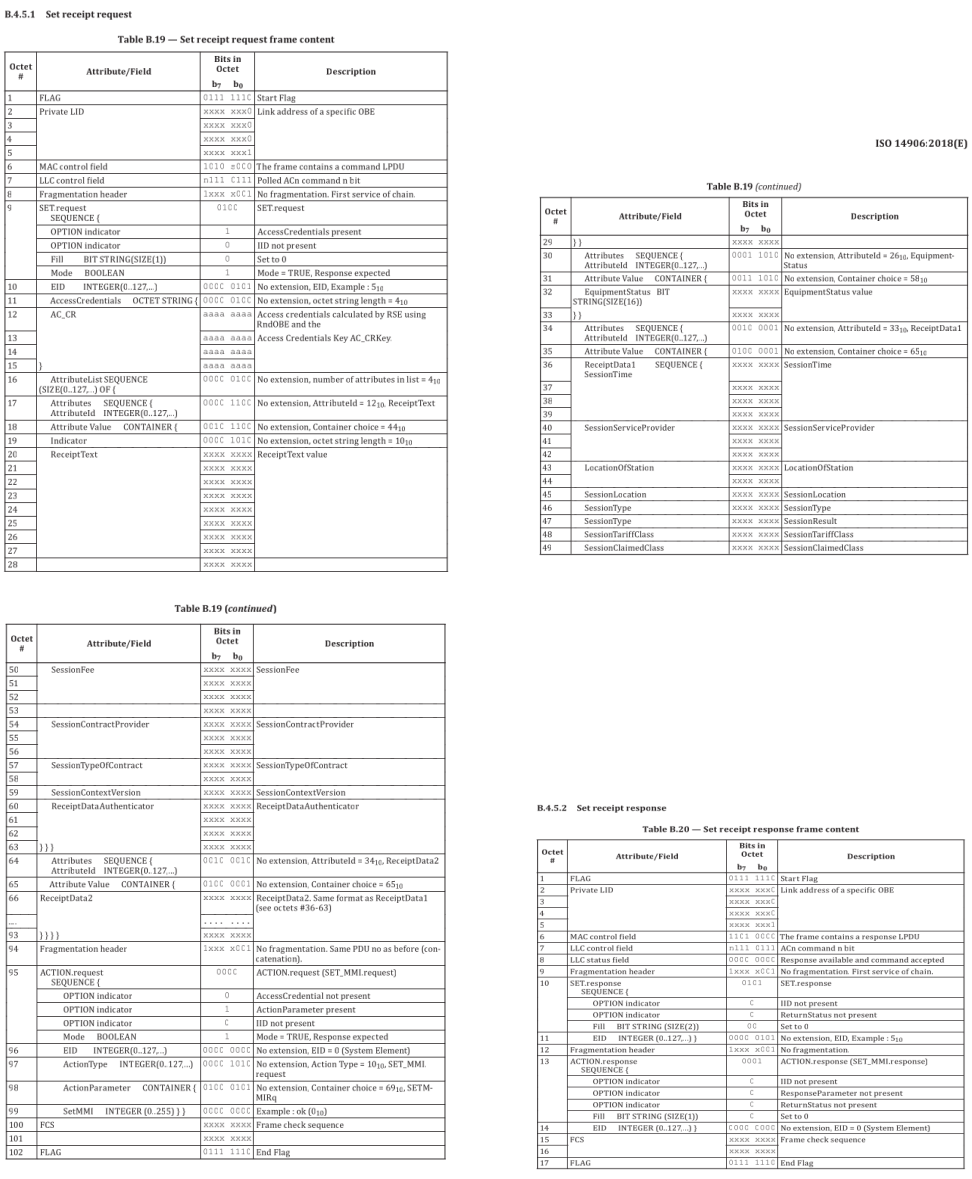
Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
88	SessionTypeOfContract	xxxx xxxx	SessionTypeOfContract
89		xxxx xxxx	
90	SessionContextVersion	xxxx xxxx	SessionContextVersion
91	ReceiptDataAuthenticator	xxxx xxxx	ReceiptDataAuthenticator
92		xxxx xxxx	
93		xxxx xxxx	
94	} } }	xxxx xxxx	
95	Attributes SEQUENCE { AttributeId INTEGER(0..127,...)	001C 001C	No extension, AttributeId = 34_{10}, ReceiptData2
96	Attribute Value CONTAINER {	010C 0001	No extension, Container choice = 65_{10}
97	ReceiptData2	xxxx xxxx	ReceiptData2. Same format as ReceiptData1 (see octets # 67-94)
...		....	
124	} } } }	xxxx xxxx	
125	FCS	xxxx xxxx	Frame check sequence
126		xxxx xxxx	
127	FLAG	0111 111C	End Flag

NOTE VehicleSpecificCharacteristics, VehicleDimensions and Vehicle Axles are not included in the examples in this table.

Рис. 6: Структура полей Presentation Response

2.1.3.5 Set Receipt Request

В случае успеха предыдущего шага система на дороге запрашивает разрешение на запись данных в транспондер (время проезда, географическое расположение рамки на дороге и т.д.)



B.4.5.1 Set receipt request

Table B.19 — Set receipt request frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 1110	Start Flag
2	Private LID	xxxx xxx0	Link address of a specific OBE
3		xxxx xxx0	
4		xxxx xxx0	
5		xxxx xxx1	
6	MAC control field	1C1O aCC0	The frame contains a command LPDU
7	LLC control field	n111 C111	Polled ACn command n bit
8	Fragmentation header	1xxx xCC1	No fragmentation. First service of chain.
9	SET.request SEQUENCE {	010C	SET.request
	OPTION indicator	1	AccessCredentials present
	OPTION indicator	0	IID not present
	Fill BIT STRING(SIZE(1))	0	Set to 0
	Mode BOOLEAN	1	Mode = TRUE, Response expected
10	EID INTEGER(0..127,...)	000C 0101	No extension, EID, Example : 5_{10}
11	AccessCredentials OCTET STRING (	000C 010C	No extension, octet string length = 4_{10}
12	AC_CR	aaaa aaaa	Access credentials calculated by RSE using RndOBE and the
13		aaaa aaaa	Access Credentials Key AC_CRKey.
14		aaaa aaaa	
15	)	aaaa aaaa	
16	AttributeList.SEQUENCE (SIZE(0..127,..) OF {	000C 01CC	No extension, number of attributes in list = 4_{10}
17	Attributes SEQUENCE { AttributeId INTEGER(0..127,..)	000C 11CC	No extension, AttributeId = 12_{10}, ReceiptText
18	Attribute Value CONTAINER {	001C 11CC	No extension, Container choice = 44_{10}
19	Indicator	000C 101C	No extension, octet string length = 10_{10}
20	ReceiptText	xxxx xxxx	ReceiptText value
21		xxxx xxxx	
22		xxxx xxxx	
23		xxxx xxxx	
24		xxxx xxxx	
25		xxxx xxxx	
26		xxxx xxxx	
27		xxxx xxxx	
28		xxxx xxxx	

Table B.19 (*continued*)

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
50	SessionFee	xxxx xxxx	SessionFee
51		xxxx xxxx	
52		xxxx xxxx	
53		xxxx xxxx	
54	SessionContractProvider	xxxx xxxx	SessionContractProvider
55		xxxx xxxx	
56		xxxx xxxx	
57	SessionTypeOfContract	xxxx xxxx	SessionTypeOfContract
58		xxxx xxxx	
59	SessionContextVersion	xxxx xxxx	SessionContextVersion
60	ReceiptDataAuthenticator	xxxx xxxx	ReceiptDataAuthenticator
61		xxxx xxxx	
62		xxxx xxxx	
63	} } }	xxxx xxxx	
64	Attributes SEQUENCE { AttributeId INTEGER(0..127,...)	001C 001C	No extension, AttributeId = 34_{10}, ReceiptData2
65	Attribute Value CONTAINER {	010C 0001	No extension, Container choice = 65_{10}
66	ReceiptData2	xxxx xxxx	ReceiptData2. Same format as ReceiptData1 (see octets #36-63)
...		....	
93	} } } }	xxxx xxxx	
94	Fragmentation header	1xxx xCC1	No fragmentation. Same PDU no as before (concatenation).
95	ACTION.request SEQUENCE {	000C	ACTION.request (SET_MMI.request)
	OPTION indicator	0	AccessCredential not present
	OPTION indicator	1	ActionParameter present
	OPTION indicator	C	IID not present
	Mode BOOLEAN	1	Mode = TRUE, Response expected
96	EID INTEGER(0..127,...)	000C 0000	No extension, EID = 0 (System Element)
97	ActionType INTEGER(0..127,...)	000C 1010	No extension, Action Type = 10_{10}, SET_MMI. request
98	ActionParameter CONTAINER {	010C 0101	No extension, Container choice = 69_{10}, SETM-MIRq
99	SetMMI INTEGER (0..255) } }	0000 0000	Example : ok (0_{10})
100	FCS	xxxx xxxx	Frame check sequence
101		xxxx xxxx	
102	FLAG	0111 1110	End Flag

ISO 14906:2018(E)

Table B.19 *(continued)*

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
29	} }	xxxx xxxx	
30	Attributes SEQUENCE { AttributeId INTEGER(0..127,...)	0001 1010	No extension, AttributeId = 26_{10}, Equipment-Status
31	Attribute Value CONTAINER {	0011 1010	No extension, Container choice = 58_{10}
32	EquipmentStatus BIT STRING(SIZE(16))	xxxx xxxx	EquipmentStatus value
33	} }	xxxx xxxx	
34	Attributes SEQUENCE { AttributeId INTEGER(0..127,...)	001C 0001	No extension, AttributeId = 33_{10}, ReceiptData1
35	Attribute Value CONTAINER {	010C 0001	No extension, Container choice = 65_{10}
36	ReceiptData1 SEQUENCE { SessionTime	xxxx xxxx	SessionTime
37		xxxx xxxx	
38		xxxx xxxx	
39		xxxx xxxx	
40	SessionServiceProvider	xxxx xxxx	SessionServiceProvider
41		xxxx xxxx	
42		xxxx xxxx	
43	LocationOfStation	xxxx xxxx	LocationOfStation
44		xxxx xxxx	
45	SessionLocation	xxxx xxxx	SessionLocation
46	SessionType	xxxx xxxx	SessionType
47	SessionType	xxxx xxxx	SessionResult
48	SessionTariffClass	xxxx xxxx	SessionTariffClass
49	SessionClaimedClass	xxxx xxxx	SessionClaimedClass

B.4.5.2 Set receipt response

Table B.20 — Set receipt response frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 1110	Start Flag
2	Private LID	xxxx xxxC	Link address of a specific OBE
3		xxxx xxxC	
4		xxxx xxxC	
5		xxxx xxx1	
6	MAC control field	1C1 OCCC	The frame contains a response LPDU
7	LLC control field	n111 C111	ACn command n bit
8	LLC status field	000C 0000	Response available and command accepted
9	Fragmentation header	1xxx xCC1	No fragmentation. First service of chain.
10	SET.response SEQUENCE {	01C1	SET.response
	OPTION indicator	C	IID not present
	OPTION indicator	C	ReturnStatus not present
	Fill BIT STRING (SIZE(2))	00	Set to 0
11	EID INTEGER (0..127,...) }	000C 0101	No extension, EID, Example : 5_{10}
12	Fragmentation header	1xxx xCC1	No fragmentation.
13	ACTION.response SEQUENCE {	0001	ACTION.response (SET_MMI.response)
	OPTION indicator	C	IID not present
	OPTION indicator	C	ResponseParameter not present
	OPTION indicator	C	ReturnStatus not present
	Fill BIT STRING (SIZE(1))	C	Set to 0
14	EID INTEGER (0..127,...) }	C00C C000	No extension, EID = 0 (System Element)
15	FCS	xxxx xxxx	Frame check sequence
16		xxxx xxxx	
17	FLAG	0111 1110	End Flag

Рис. 7: Структура полей Set Receipt Request

2.1.3.6 Set Receipt Response

Транспондер отвечает о том, возможна ли запись

B.4.5.2 Set receipt response

Table B.20 — Set receipt response frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 111C	Start Flag
2	Private LID	xxxx xxxC	Link address of a specific OBE
3		xxxx xxxC	
4		xxxx xxxC	
5		xxxx xxx1	
6	MAC control field	11C1 0CCC	The frame contains a response LPDU
7	LLC control field	n111 C111	ACn command n bit
8	LLC status field	CCCC CCCC	Response available and command accepted
9	Fragmentation header	1xxx xCC1	No fragmentation. First service of chain.
10	SET.response SEQUENCE {	0101	SET.response
	OPTION indicator	C	IID not present
	OPTION indicator	C	ReturnStatus not present
	Fill BIT STRING (SIZE(2))	CC	Set to 0
11	EID INTEGER (0..127,...) }	CCCC C101	No extension, EID, Example : 5_{10}
12	Fragmentation header	1xxx xCC1	No fragmentation.
13	ACTION.response SEQUENCE {	0001	ACTION.response (SET_MMI.response)
	OPTION indicator	C	IID not present
	OPTION indicator	C	ResponseParameter not present
	OPTION indicator	C	ReturnStatus not present
	Fill BIT STRING (SIZE(1))	C	Set to 0
14	EID INTEGER (0..127,...) }	CCCC CCCC	No extension, EID = 0 (System Element)
15	FCS	xxxx xxxx	Frame check sequence
16		xxxx xxxx	
17	FLAG	0111 111C	End Flag

Рис. 8: Структура полей Set Receipt Response

2.1.3.7 Echo Request

Далее происходит уже формальный запрос (эхо) от системы на дороге, для того, чтобы закрыть соединение

B.4.6.1 Tracking request (ECHO.request)

Table B.21 — Tracking request frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 111C	Start Flag
2	Private LID	xxxx xxx0	Link address of a specific OBE
3		xxxx xxx0	
4		xxxx xxx0	
5		xxxx xxx1	

Table B.21 *(continued)*

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
6	MAC control field	1C10 sCC0	The frame contains a command LPDU
7	LLC control field	n111 C111	Polled ACn command n bit
8	Fragmentation header	1xxx xCC1	No fragmentation.
9	ACTION.request SEQUENCE {	0000	ACTION.request (ECHO.request)
	OPTION indicator	0	No Access Credentials
	OPTION indicator	1	ActionParameter present
	OPTION indicator	0	IID not present
	Mode BOOLEAN	1	Mode = TRUE, Response expected
10	EID INTEGER (0..127,...)	0000 0000	No extension, EID = 0 (System Element)
11	ActionType INTEGER (0..127,...)	0000 1111	No extension, Action Type = 15_{10}, ECHO.request
12	ActionParameter CONTAINER {	0000 001C	No extension, Container Choice = 2_{10}, Octet string
13	} }	0000 0000	No extension. String length = 0 octets
14	FCS	xxxx xxxx	Frame check sequence
15		xxxx xxxx	
16	FLAG	0111 111C	End Flag

Рис. 9: Структура полей Echo Request

2.1.3.8 Echo Response

Ответ траснпондера:

Table B.22 — Tracking response frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 111C	Start Flag
2	Private LID	xxxx xxx0	Link address of a specific OBE
3		xxxx xxx0	
4		xxxx xxx0	
5		xxxx xxx1	
6	MAC control field	11C1 0CC0	The frame contains a response LPDU
7	LLC control field	n111 C111	ACn command n bit
8	LLC status field	0CCC 0CCC	Response available and command accepted
9	Fragmentation header	1xxx xCC1	No fragmentation.
10	ACTION.response SEQUENCE {	0001	ACTION.response (ECHO.response)
	OPTION indicator	0	No IID
	OPTION indicator	1	ResponseParameter present
	OPTION indicator	0	ReturnStatus not present
	Fill BIT STRING (SIZE(1))	0	Set to 0.
11	EID INTEGER (0..127,...)	0000 0000	No extension, EID = 0 (System Element)
12	ResponseParameter CONTAINER {	0000 001C	No extension, Container Choice = 2_{10}, Octet string

Table B.22 *(continued)*

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
13	} }	0000 0000	No extension. String length = 0 octets
14	FCS	xxxx xxxx	Frame check sequence
15		xxxx xxxx	
16	FLAG	0111 111C	End Flag

2.1.3.9 Closing

Закрытие соединения:

B.4.6.3 Closing

Table B.23 — Closing frame content

Octet #	Attribute/Field	Bits in Octet b_7 b_0	Description
1	FLAG	0111 111C	Start Flag
2	Private LID	xxxx xxx0	Link address of a specific OBE
3		xxxx xxx0	
4		xxxx xxx0	
5		xxxx xxx1	
6	MAC control field	10CC 00CC	The frame contains a command LPDU
7	LLC control field	0000 0011	UI command
8	Fragmentation header	1xxx x0C1	No fragmentation.
9	EVENT_REPORT.request SEQUENCE {	0010	EVENT_REPORT.request (RELEASE)
	OPTION indicator	C	AccessCredential not present
	OPTION indicator	C	EventParameter not present
	OPTION indicator	C	IID not present
	Mode BOOLEAN	C	Mode = FALSE, No response expected
10	EID INTEGER (0..127,...)	0000 0000	No extension, EID = 0 (system element)
11	EventType INTEGER (0..127,...) }	0000 0000	No extension, Event Type = 0, RELEASE
12	FCS	xxxx xxxx	Frame check sequence
13		xxxx xxxx	
14	FLAG	0111 1110	End Flag

Рис. 11: Структура полей Echo Request

2.2 Обзор аналогов

В данном разделе приведен обзор системы компании Norbit, которая и стала отправной точкой для создания россейсеого аналога

1. ITS NORBIT (Система умного планирования дорожного трафика) [6] — программно-аппаратный комплекс, который осуществляет полную обработку данными в ходе транзакции между RSU и OBU

Norbit являлся мнополистом до ухода из России, поэтому его решения были взяты за основу для создания российского аналога

2.3 Выбор окружения для разработки

После обсуждения в команде разработчков было решено использовать язык C++ 11 версии, в виду необходимости быстрой работы с поступающими данными, а также необходимостью написания функциональности аппаратного обеспечения. В качестве ОС была выбрана Ubuntu 22.04 как ОС семейства GNU/Linux.

3 Реализация

3.1 Входные данные с канального уровня

С канального уровня на прикладной уровень поступает строка в hex формате, которую в дальнейшем необходимо преобразовать в бинарный формат, поскольку того требует спецификация стандартов, использованных при разработке. Пример реальной строки в hex формате:
100a002c 00600000 24760e71 c0039190 0001c101 02105700 01ff0070 02021dd1 0204118e 0d7bf301 00320100 00320100.
Здесь первые 2 октета - заголовок, а последний - незначащий, таким образом полезной нагрузкой являются 9 октетов. Для преобразования в бинарный формат используется следующая функция:

```
    std::string hextobin(const std::string &s)
{
    std::string out;
    for (auto &i : s)
    {
        uint8_t n;
        if (i ≤'9' and i ≥'0')
            n = i - '0';
        else
            n = 10 + i - 'A';
        for (int8_t j = 3; j ≥0; --j)
            out.push_back((n & (1 << j)) ? '1' : '0');
    }

    return out;
}
```

После этого строка преобразуется в строку из 0 и 1, после чего уже происходит её разбор согласно структуре, указаной на Рис.3

3.2 Выходные данные для протокола EARP

Протокол EARP (EFC Read Attribute Protocol - протокол чтения аттрибутов) требует специальный формат вывода, не соответствующий приходящему в него бинарному формату. Структура должна иметь следующий вид:

Part	Len	Description
Timestamp	19	Timestamp for the transaction, according to the RSUs internal clock
(space)	1	A single space character as delimiter
ContextMark	(40)	The OBU ContextMark, consisting of subfields:
CountryCode	3	The ISO-3166-1 country code of the OBU ContextMark
,	1	A single comma character as delimiter
OperatorId	5	The OperatorId part of the OBU ContextMark
,	1	A single comma character as delimiter
TypeOfContract	5	The TypeOfContract part of the OBU ContextMark
,	1	A single comma character as delimiter
ContextVersion	3	The ContextVersion part of the OBU ContextMark
:	1	A single colon character as delimiter
VST Parameter	(20)	DSRC application type dependent 'extra' information associated with the ContextMark. May be e.g. nonce value
(space)	1	A single space character
OBU-Info	(14)	Info about the OBU, as contained in the VST. consisting of the subfields:
EquipmentClass	4	The OBU EquipmentClass (in hexadecimal!)
/	1	Slash character as delimiter
ManufacturerId	4	The OBU ManufacturerId (in hexadecimal!)
/	1	Slash character as delimiter
ObeStatus	4	OBU Status (in hexadecimal!)
(space)	1	A single space character
Non-OBU Info	(28)	Information not read from the OBU, consisting of the subfields:
[	1	A left square bracket
ComputerNo	3	The number of the RSU performing the transaction
/	1	Slash character as delimiter

Part	Len	Description
Speed	6	The estimated speed of the vehicle, in km/h. Negative number means the vehicle is driving in the wrong direction. Speed is given with one decimal digit. Speed will always contain a sign (plus or minus).
/	1	Slash character as delimiter
Xpos	4	The estimated position of the OBU along the road
/	1	Slash character as delimiter
Ypos	4	The estimated position of the OBU across the road
]	1	A right square bracket
(space)	1	A single space character

Рис. 12: Структура данных в формате EARP

Пример вывода информации в протоколе EARP:

```
20100611T095008.513 578,00008,00001,001:--------------------
0000/002a/0000
[104/00645 +015.7/05.1/00.0]
AutoPASS <0EC7AC9DB0EA114C> |
099:021830C008CC3CDE140000C28422C6DF9C187A21EDF42A770F2B |
```

Рис. 13: Выходные данные

3.2.1 Время для протокола EARP

Как видно в примере выше, время необходимо не в самом типичном формате. Для создания строки, содержащее время в необходимом формате реализована следующая функция:

```
char *current_time()
{
    char *currentTime = (char *)malloc(sizeof(char) * 20);
    timeval curTime;
    gettimeofday(&curTime, NULL);
    int millisecs = curTime.tv_usec / 1000;

    char current_time_no_mill[16];
    strftime(current_time_no_mill, 16, ''%Y%m%dT%H%M%S'',
       localtime(&curTime.tv_sec));
    sprintf(currentTime, ''%s.%03d'', current_time_no_mill, millisecs);
    return currentTime;
}
```

3.3 Проверка подлинности ключей

На этапе запроса данных с транспондера (Presentation Request) необходимо провести вычисления алгоритмом 3DES. RSE занимается вычислениями независимо от OBU, используя только ключи, предоставленные в ходе обмена с ним, поэтому необходимо реализовать и шифрование на RSE и сверить с результатами, которые ждёт OBU. В случае несовпадения значений (где значения это HEX строки) обмен данными прекращается и транзакция считается неудачной. Код реализации 3DES слишком велик, поэтому в листинге ниже приведены совпадения по вводимым и выводимым значениям, согласно примерам из документации:

```
uint64_t AC_CR-KeyReference = 0x1234123412341234;
uint64_t key[3] ={0x5757575757575757, 0xEFEFEFEFEFEFEFEF,
   0x5757575757575757};
uint64_t result = AC_CR-KeyReference;
```

```
cout<<''\Шифрованиеn \ и\ дешифрование DES & 3-DES \n'';
printf(''AC_CR-KeyReference: %016llX\n'', AC_CR-KeyReference);
result = des(AC_CR-KeyReference, key[0], 'e');
printf(''Des: %016llx\n'', result);
result = des(result, key[1], 'd');
result = des(result, key[2], 'e');
printf(''3Des: %016llx\n'', result);
```

Получившиеся этапы шифрования и дешифрования:

```
AC_CR-KeyReference: 1234123412341234
Des: a1787bb095069cee
3Des: 9b48aae07a7bc008
```

Вот результаты вычисления с теми же значениями из официальной документации (документ [3]:

G.4.2 Access Credentials Key

EXAMPLE The derivation of the Access Credentials Key is quite similar to the derivation of the Authenticators keys. Instead of the PAN the AC_CR-KeyReference is used.

Using the following application data values and Master Key:

— AC_CR-KeyReference: '12 34'

— MAcK: '57 57 57 57 57 57 57 57 EF EF EF EF EF EF EF EF'

this gives:

VAL = 'AC_CR-KeyReference || AC_CR-KeyReference || AC_CR-KeyReference || AC_CR-KeyReference' = '12 34 12 34 12 34 12 34'

and:

AcK = ede[MAcK](VAL) = '9B 48 AA E0 7A 7B C0 08'

Again, a different AC_CR-KeyReference or Master Key will produce a completely different Access Credentials Key.

Рис. 14: Пример вычислений ключей из стандарта

Эти вычисления в том числе использовались для формирования этапов.

Ниже представлен код примера формирования данных для Presentation Request:

```
    std::string RndRSE_gen()
{
```

```
    time_t now = time(0);
    tm *ltm = localtime(&now);
    std::string bin_year = (std::bitset<7>(ltm→tm_year -
       90)).to_string();
    std::string bin_month = std::bitset<4>(1 + ltm→tm_mon).to_string();
    std::string bin_day = std::bitset<5>(ltm→tm_mday).to_string();
    std::string bin_hour = std::bitset<5>(ltm→tm_hour).to_string();
    std::string bin_min = std::bitset<6>(ltm→tm_min).to_string();
    std::string bin_sec = std::bitset<5>(ltm→tm_sec).to_string();
    return bin_year + bin_month + bin_day + bin_hour + bin_min + bin_sec;
}
std::string Pres_Req_create_bit()
{
    std::string start_flag = ''01111110'';
    std::string LID = ''11101000100111101001101000011001'';
    std::string MAC_control_field = ''10101000'';
    std::string LLC_control_field = ''01110111'';
    std::string Fragmentation_header_1 = ''11110001'';
    std::string GET_STAMPED_req = ''0000'';
    std::string AcCred_1 = ''1'';
    std::string ActParam = ''1'';
    std::string IID_1 = ''0'';
    std::string Mode = ''1'';
    std::string EID = ''00000001'';
    std::string Action_type = ''00000000'';
    std::string AcCred_Length = ''00000100'';
    std::string AC_CR = hextobin(gen());
    std::string ActionParameter = ''00010001'';
    std::string AttributeList_number_1 = ''00000001'';
    std::string AttributeId = ''001000000''; // == 32_10 == PaymentMeans
    std::string RndRSE = RndRSE_gen();
    std::string KeyRef_Op = ''00000000'';
    std::string Fragmentation_header_2 = ''11110001'';
    std::string GET_request = ''0110'';
    std::string AcCred_2 = ''1'';
    std::string IID_2 = ''0'';
    std::string AttrIdList = ''1'';
    std::string Fill = ''0'';
    std::string AttributeList_number_2 = ''00000110'';
    std::string VehicleLicensePlateNumber = ''00010000'';
```

```
    std::string VehicleClass = ''00010001'';       // 17_10
    std::string VehicleWeightLimits = ''00010100''; // 20_10
    std::string EquipmentStatus = ''00011010'';    // 26_10
    std::string ReceiptData_1 = ''00100001'';
    std::string ReceiptData_2 = ''00100010'';
    std::string FCS = ''0000000011111111'';
    std::string End_Flag = ''01111110'';

    return start_flag + LID + MAC_control_field + LLC_control_field +
       Fragmentation_header_1 + GET_STAMPED_req + AcCred_1 + ActParam +
       IID_1 + Mode + EID +
           Action_type + AcCred_Length + AC_CR + ActionParameter +
              AttributeList_number_1 + AttributeId + RndRSE + KeyRef_Op +
              Fragmentation_header_2 + GET_request +
           AcCred_2 + IID_2 + AttrIdList + Fill + AC_CR +
              AttributeList_number_2 + VehicleLicensePlateNumber +
              VehicleClass + VehicleWeightLimits +
           EquipmentStatus + ReceiptData_1 + ReceiptData_2 + FCS +
              End_Flag;
}
```

Всё это в итоге превращается в бинарную строку, которая далее уже анализируется на стороне OBU.

Заключение

В результате работы были выполнены следующие задачи.

- Проанализированы имеющуюся документацию предыдущих производителей на российском рынке
- Выполнен обзор соответствующие стандарты необходимые для разработки системы
- Реализована выдачу информации в режиме реального времени для информирования оператора о текущем статусе в терминах описанных в документах протоколов
- Реализовано логирование работы системы для анализа ошибок и сбора статистики
- Реализована часть обмена информации между канальным и прикладным уровнями
- Реализовано шифрование алгоритмом 3DES, которое повторяет результаты примеров из официальной документации

К сожалению, работа не была завершена полностью, не была проведена апробация и отправка в биллинг, поскольку проект был закрыт до этих этапов.

Исходный код находится на локальном сервере Мобил-Групп на платформе Gitlab [2].

Список литературы

[1] BS-EN 12834. — URL: https://www.en-standard.eu/bs-en-12834-2003-road-transport-and-traffic-telematics-dedicated- (дата обращения: 2024-02-29).

[2] Gitlab. — URL: https://gitlab.com/ (дата обращения: 2024-02-29).

[3] ISO14906. — URL: https://www.iso.org/standard/82931.html (дата обращения: 2024-02-29).

[4] Mobil-Group. — URL: https://mobil-group.spb.ru/ (дата обращения: 2023-12-15).

[5] ЛИС. — URL: https://labics.ru/ (дата обращения: 2023-12-15).

[6] Норбит. — URL: https://norbit.com/its/products/ (дата обращения: 2024-01-08).